AQUACOLD: A Novel Crowdsourced Linked Data Question Answering System

Nicholas W. Collis[a], Ingo Frommholz[b]

[a]*Institute for Research in Applicable Computing, University of Bedfordshire, Luton, England*
[b]*School of Mathematics and Computer Science, University of Wolverhampton, UK*

Abstract
Question Answering (QA) systems provide answers to Natural Language (NL) questions posed by humans. The Linked Data (LD) web provides an ideal knowledge base for QA as the framework expresses structure and relationships between data which assist in question parsing. Despite this, recent attempts at NL QA over LD struggle when faced with complex questions due to the challenges in automatically parsing NL into a structured query language, forcing end users to learn languages such as SPARQL which can be challenging for those without a technical background. There is a need for a system which returns accurate answers to complex natural language questions over linked data, improving the accessibility of linked data search by abstracting the complexity of SPARQL whilst retaining its expressivity. This work presents AQUACOLD (Aggregated Query Understanding And Construction Over Linked Data) a novel LD QA system which harnesses the power of crowdsourcing to meet this need. AQUACOLD uses query templates built by other users to answer questions which enables the system to handle queries of significant complexity. This paper provides an overview of the system and presents the results of a technical and user evaluation against the QALD-9 benchmark.

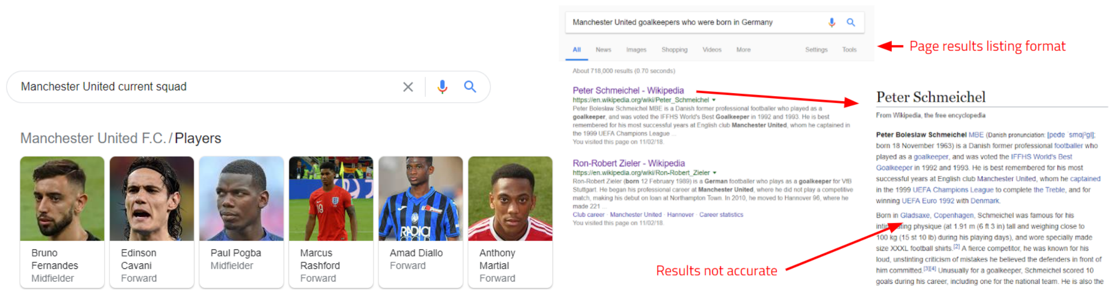
Keywords
Linked Data, Natural Language, Question Answering, Crowdsourcing, SPARQL

1. Introduction

Question Answering is a subset of the Natural Language Processing and Information Retrieval fields that focuses on providing direct answers to end users in response to a question formed of natural language [1]. This method of providing direct answers to questions is preferred by users over providing links to other information sources that may or may not contain those answers [2]. Modern search engines and question answering systems including Google and Bing often return direct answers to simple queries such as *'Who is in the current Manchester United squad?'* (see figure 1).

However, simple, common queries like this form the minority of searches as 97% of search engine queries have been shown to occur ten times or less [3]. Search Engines and Question Answering systems often fail with queries of greater complexity, providing instead links to websites which may be incorrect (see Figure1). This highlights the need for a system which can understand more complex queries and return accurate answers.

BIRDS 2021: Bridging the Gap between Information Science, Information Retrieval and Data Science, March 19 2021, Online
nwcphd@gmail.com (N. W. Collis); ifrommholz@acm.org (I. Frommholz)



Manchester United current squad

Manchester United F.C. / Players

Bruno Fernandes Midfielder
Edinson Cavani Forward
Paul Pogba Midfielder
Marcus Rashford Forward
Amad Diallo Forward
Anthony Martial Forward

Manchester United goalkeepers who were born in Germany

About 718,000 results (0.70 seconds)

Peter Schmeichel - Wikipedia
https://en.wikipedia.org/wiki/Peter_Schmeichel
Peter Boleslaw Schmeichel MBE is a Danish former professional footballer who played as a **goalkeeper**, and was voted the IFFHS World's Best **Goalkeeper** in 1992 and 1993. He is best remembered for his most successful years at English club **Manchester United**, whom he captained in the 1999 UEFA Champions League ...
You visited this page on 11/02/18.

Ron-Robert Zieler - Wikipedia
https://en.wikipedia.org/wiki/Ron-Robert_Zieler
Ron-Robert Zieler (**born** 12 February 1989) is a **German** footballer who plays as a **goalkeeper** for VfB Stuttgart. He began his professional career at **Manchester United**, where he did not play a competitive match, making his debut on loan at Northampton Town. In 2010, he moved to Hannover 96, where he made 221 ...
Club career · Manchester United · Hannover · Career statistics
You visited this page on 11/02/18.

Peter Schmeichel

From Wikipedia, the free encyclopedia

Peter Bolesław Schmeichel MBE (Danish pronunciation: [pedɐ ˈsmajˀɡl]; born 18 November 1963) is a Danish former professional footballer who played as a goalkeeper, and was voted the IFFHS World's Best Goalkeeper in 1992 and 1993. He is best remembered for his most successful years at English club Manchester United, whom he captained in the 1999 UEFA Champions League to complete the Treble, and for winning UEFA Euro 1992 with Denmark.

Born in Gladsaxe, Copenhagen, Schmeichel was famous for his inti[illegible]ting physique (at 1.91 m (6 ft 3 in) tall and weighing close to 100 kg (15 st 10 lb) during his playing days), and wore specially made size XXXL football shirts.[2] A fierce competitor, he was known for his loud, unstinting criticism of mistakes he believed the defenders in front of him committed.[3][4] Unusually for a goalkeeper, Schmeichel scored 10 goals during his career, including one for the national team. He is also the

Figure 1: (L) Direct answers given to simple question posed in a search engine (R) A more complex question failing to return direct answers, with inaccurate web listings given instead

To overcome the challenges present in parsing unstructured text, many QA systems reason over a structured data source such as the Linked Data Web instead of, or in conjunction with, unstructured text to return an answer [4].

Linked Data — described by creator Sir Tim Berners-Lee as *'The Semantic Web Done Right'* [5] — refers to a technology stack (RDF, OWL, SPARQL) that provides a mechanism for data to be published, queried and inferred on the web [6]. Using URIs (Uniform Resource Identifiers) as unique identifiers, data can be consistently referenced by programs from different domains to foster interoperability. The additional inherent structure can provide more accurate results for *factoid* questions (*what, where, when, which, who,* or *is*) than unstructured text [7].

QA systems which harness Linked Data typically decompose natural language queries into a structured query language such as SPARQL [8]. These approaches are successful when answering questions with high complexity (depth) over a narrow domain (breadth) or conversely, shallow depth over a broad domain, but are less effective with increased question depth and breadth, due the challenges of producing query representations for every question type [9].

Previous work has shown that harnessing crowdsourced workers to translate natural language to SPARQL can resolve this problem, resulting in a QA space of greater depth and breadth than is possible with programmatic methods [10]. However, much of this work has relied on paid microtask workers to translate between natural language queries and linked data representation, which can result in biased results [11], misaligned incentives [12] and is not easily scalable.

AQUACOLD is a novel Question Answering system which fills this gap by allowing users to both get answers to and produce answers for Natural Language questions using a recognisable spreadsheet-style UI for creating and filtering linked data sets, which can be labelled with natural language, transformed into templates to answer related questions, and the answers retrieved in response to a natural language question. This allows complex questions to be asked and answered by the crowd over a wide domain of knowledge with organically aligned incentives that arise from genuine information need.

This paper outlines AQUACOLD and its performance against the QALD-9 benchmark and is structured as follows. In section 2 we discuss related work. In section 3 we introduce the AQUACOLD system. Section 4 outlines the evaluation methodology used to assess the effectiveness of the system, section 5 reports the results of the evaluation and section 6 summarises the findings and outlines future work.

2. Related Work

Tools that provide answers to questions from Linked Data sources are typically classified into Natural Language Interfaces for Linked Data (NLI-LD) where the query is entered as natural language text and Query Builder Interfaces for Linked Data (QB-LD) where the query is expressed through manipulation of an interface.

NLI-LD is rooted in field of natural language search over structured databases, which dates back to search systems for specific domains such as baseball [13] and lunar rocks [14], leading to representation languages which interpret the question independent of the database structure, as seen in MASQUE/SQL [15], an early NL front-end to an SQL database.

Recent NLI-LD systems use a combination of techniques to convert natural language into a structured query language. Some systems restrict the user to controlled natural language input that requires specific words used in a specific order [2] whereas others allow complex sentence structures [16]. Many NLI-LD systems decompose natural language queries into SPARQL using generic query templates [17, 18]. This approach has proved successful for search over closed domains, but less effective over open domains due the challenges of designing appropriate templates for every possible question [9].

Linked Data Query Builder interfaces (QB-LD) abstract the underlying SPARQL code into a graphical representation which can be manipulated by the user to form a query. Some QB-LD systems use graph interaction paradigms, such as NITELITE [19] and Semantic Crystal [20] as these align well with the graph based nature of RDF data. Other QB-LD systems follow a tabular structure for representing and interacting with Linked Data results and/or queries, such as FalconS [21] and Freebase Parallax [22]. Many QB-LD systems [23, 24, 25] allow the user to progressively refine their query through the application of predicate and value pairs termed *facets*[26]. Facets have been shown to benefit users composing exploratory queries over Linked Data [27]. QB-LD systems offer a greater robustness than NLI-LD as it is clear when an answer cannot be found due to lack of data as opposed to incorrect query syntax, but are limited in flexibility as the progressive filtering approach necessitates simpler queries.

Hybrid approaches combine elements of NLI-LD and QB-LD, using the readability of natural language as an entry point for a query with the query building elements highlighting what terminology can or cannot be used. Examples of such hybrid systems are SPARKLIS [28], CODE: Linked Data Query Wizard [29], Atomate [30] and Ginseng [31].

Some systems incorporate machine learning techniques into NLI-LD. One high profile example being the success of DeepQA, the technology behind IBM Watson [32], which was able to win a game of the TV show *Jeopardy!* against expert human opponents by using a variety of machine learning and Natural Language Processing techniques to compute the correct answer over range of structured and unstructured data sources, including Linked Data.

Incorporating crowdsourcing into Linked Data search remains an active area of research [33]. Some predict the combination of the open, interconnected Linked Data web and the power of the crowd will produce a *'Global Brain Semantic Web'* [2]. Crowdsourcing has been used to enhance many aspects of Linked Data systems, including accuracy [34], adding additional context [35], query optimisation [36], query expansion and query understanding [37]. Using the crowd to help translate natural language queries into SPARQL has shown to resolve some of the problems inherent in purely programmatic/algorithmic methods [10].

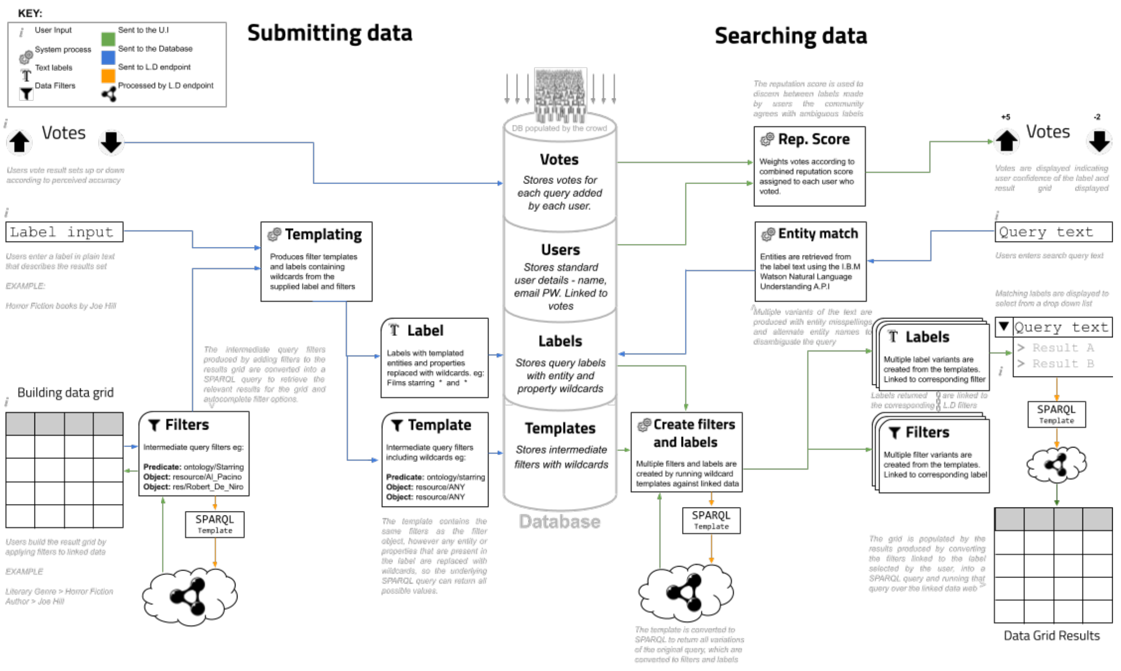
CrowdQ [37] is of particular relevance to this work, as the system employs the crowd to build query templates which can be used to answer multiple variants of the original query. CrowdQ limits the complexity of questions that can be answered to 1 semantic '*hop*'. So *'all students in class CIS01b'* would be answerable but *'birth dates of all professors who have students in class CIS01b'* would not.

A hybrid QB/NLI/CS approach has been explored in Google Squared [38] which allowed users to label 'squares' of related information. This employed data scraped from unstructured text and HTML tables and did not exploit the rich relations available from RDF Linked Data.

3. AQUACOLD System Overview

AQUACOLD (**A**ggregated **Q**uery **U**nderstanding and **C**onstruction **O**ver **L**inked **D**ata) combines the principles of Natural Language Interfaces, Query Builders and Crowdsourcing into a Question Answering system for Linked Data which requires no prior knowledge of SPARQL or the underlying data schema.

AQUACOLD allows users to find answers to questions from a Linked Data endpoint by manipulating a faceted tabular interface similar to that found in spreadsheet applications. These filter sets can then be labelled using natural language and turned into templates which allow similar questions to also be answered. Future users can then search for and retrieve these answers using natural language, vote on the quality of the results and adapt them to produce their own labeled result sets and templates. The full system diagram is shown in fig. 2.



KEY:
User Input
System process
Text labels
Data Filters
Sent to the U.I
Sent to the Database
Sent to L.D endpoint
Processed by L.D endpoint
Submitting data
Searching data
Votes
Users vote result sets up or down according to perceived accuracy
Label input
Users enter a label in plain text that describes the results set
EXAMPLE:
Horror Fiction books by Joe Hill
Templating
Produces filter templates and labels containing wildcards from the supplied label and filters
The intermediate query filters produced by adding filters to the results grid are converted into a SPARQL query to retrieve the relevant results for the grid and autocomplete filter options.
Label
Labels with templated entities and properties replaced with wildcards. eg: Films starring * and *
Building data grid
Filters
Intermediate query filters eg:
Predicate: ontology/Starring
Object: resource/Al_Pacino
Object: res/Robert_De_Niro
SPARQL Template
Template
Intermediate query filters including wildcards eg:
Predicate: ontology/starring
Object: resource/ANY
Object: resource/ANY
Users build the result grid by applying filters to linked data
EXAMPLE
Literary Genre > Horror Fiction
Author > Joe Hill
The template contains the same filters as the filter object, however any entity or properties that are present in the label are replaced with wildcards, so the underlying SPARQL query can return all possible values.
DB populated by the crowd
Votes
Stores votes for each query added by each user.
Users
Stores standard user details - name, email PW. Linked to votes
Labels
Stores query labels with entity and property wildcards
Templates
Stores intermediate filters with wildcards
Database
The reputation score is used to discern between labels made by users the community agrees with ambiguous labels
Rep. Score
Weights votes according to combined reputation score assigned to each user who voted.
+5
Votes
-2
Votes are displayed indicating user confidence of the label and result grid displayed
Entity match
Entities are retrieved from the label text using the I.B.M Watson Natural Language Understanding A.P.I
Query text
Users enters search query text
Multiple variants of the text are produced with entity misspellings and alternate entity names to disambiguate the query
Matching labels are displayed to select from a drop down list
Query text
> Result A
> Result B
Labels
Multiple label variants are created from the templates. Linked to corresponding filter
Labels returned are linked to the corresponding L.D filters
Filters
Multiple filter variants are created from the templates. Linked to corresponding label
Create filters and labels
Multiple filters and labels are created by running wildcard templates against linked data
SPARQL Template
The template is converted to SPARQL to return all variations of the original query, which are converted to filters and labels
The grid is populated by the results produced by converting the filters linked to the label selected by the user, into a SPARQL query and running that query over the linked data web
SPARQL Template
Data Grid Results

Figure 2: AQUACOLD System Architecture

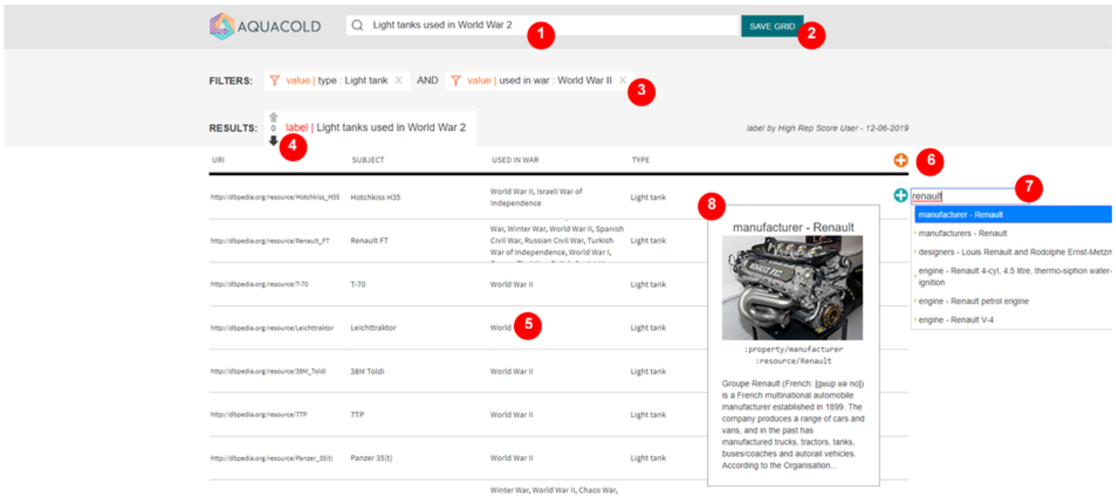
3.1. System Interface

AQUACOLD combines several features into one novel question answering tool: **natural language search**, **crowdsourced query labelling** and **voting**, a **query builder interface** and **query templating** system.

Figure 3 shows a screenshot of the AQUACOLD User Interface. The key components are a combined search and labelling box (item 1 in the figure) and Query Builder (items 3-8) which incorporates a results grid and set of customisable filters (item 3) which can explore a given linked data source, retrieving LD node labels and linking to related nodes, progressively building up a result set based on the filters entered by the user. Once complete, the result grid and associated filters can be labeled using natural language (item 1 in the figure) with autocompletion suggestions for wording and terminology based on the labels entered by other users.

As users compose natural language queries, they are guided using autocompletion (item 7) which indicates the data available from the linked data source and their labels, alongside other useful information such as images and description (item 8). Finally, a voting system (item 4) is provided to rank the results, allowing the most accurate results sets to appear first to users (item 1).

Query templates are produced based on pattern matching between related query and entity labels. These templates are used to answer similar natural language queries entered by future users, returning results from the linked data web to the results grid.


AQUACOLD
Light tanks used in World War 2
SAVE GRID
FILTERS:
value | type : Light tank
AND
value | used in war : World War II
RESULTS:
0
label | Light tanks used in World War 2
label by High Rep Score User - 12-06-2019
renault
manufacturer - Renault
manufacturers - Renault
designers - Louis Renault and Rodolphe Ernst-Metzm
engine - Renault 4-cyl, 4.5 litre, thermo-siphon water-ignition
engine - Renault petrol engine
engine - Renault V-4
manufacturer - Renault
:property/manufacturer
:resource/Renault
1
2
3
4
5
6
7
8

URI	SUBJECT	USED IN WAR	TYPE
http://dbpedia.org/resource/Hotchkiss_H35	Hotchkiss H35	World War II, Israeli War of Independence	Light tank
http://dbpedia.org/resource/Renault_FT	Renault FT	War, Winter War, World War II, Spanish Civil War, Russian Civil War, Turkish War of Independence, World War I,	Light tank
http://dbpedia.org/resource/T-70	T-70	World War II	Light tank
http://dbpedia.org/resource/Leichttraktor	Leichttraktor	World	Light tank
http://dbpedia.org/resource/38M_Toldi	38M Toldi	World War II	Light tank
http://dbpedia.org/resource/7TP	7TP	World War II	Light tank
http://dbpedia.org/resource/Panzer_35(t)	Panzer 35(t)	World War II	Light tank
		Winter War, World War II, Chaco War,	

Groupe Renault (French: [ɡʁup ʁə'no]) is a French multinational automobile manufacturer established in 1899. The company produces a range of cars and vans, and in the past has manufactured trucks, tractors, tanks, buses/coaches and autorail vehicles. According to the Organisation...

1 **Natural Language search & Filter set labelling box** – Users can type a natural language query here and also label a set of filters.

2 **Save grid button** – On clicking this button, the filters selected by the user are saved with the label provided by the user.

3 **Filters applied** – The filters selected by the user for this query.

4 **Label and voting controls** - The label given to these filters / results set and controls to add a positive or negative vote.

5 **Results grid** – Lists the results for this query which can be manipulated by applying filters.

6 **Property filter**– Users can add additional properties to the results grid here.

7 **Property value filter**– Users can add additional property value filters to the results grid by searching for them here.

8 **URI preview** – A preview window displaying the full URI, description and thumbnail graphic (from Dbpedia) of the URI currently selected (if available).

Figure 3: The AQUACOLD interface with key

3.2. The Linked Data Feedback Loop

The following scenario demonstrates how a user may engage with the system (fig. 4).

1. **User A** arrives at the site looking for information on the tanks used in World War 1. He/she enters *'Tanks used in World War 1'* into the search box. No results are found.
2. Unable to find existing results, **User A** builds the results grid by adding filters `dbp:type=:Tank` and `dbp:usedInWar=:worldWar1`.
3. The results grid complete, **User A** adds the label *'Tanks used in World War 1'*.
4. AQUACOLD populates the database with **User A's** query label and the associated SPARQL code from the results grid, together with possible variations formed by replacing all entities in the query with wildcards e.g. *Tanks used in [*]*.
5. Another user, **User B** searches for *'Tanks used in World War 2'*. Although a result grid for this query has not been explicitly created by another user, results are returned by the template created in step 4, substituting the entity *'World War 1'* with *'World War 2'*.

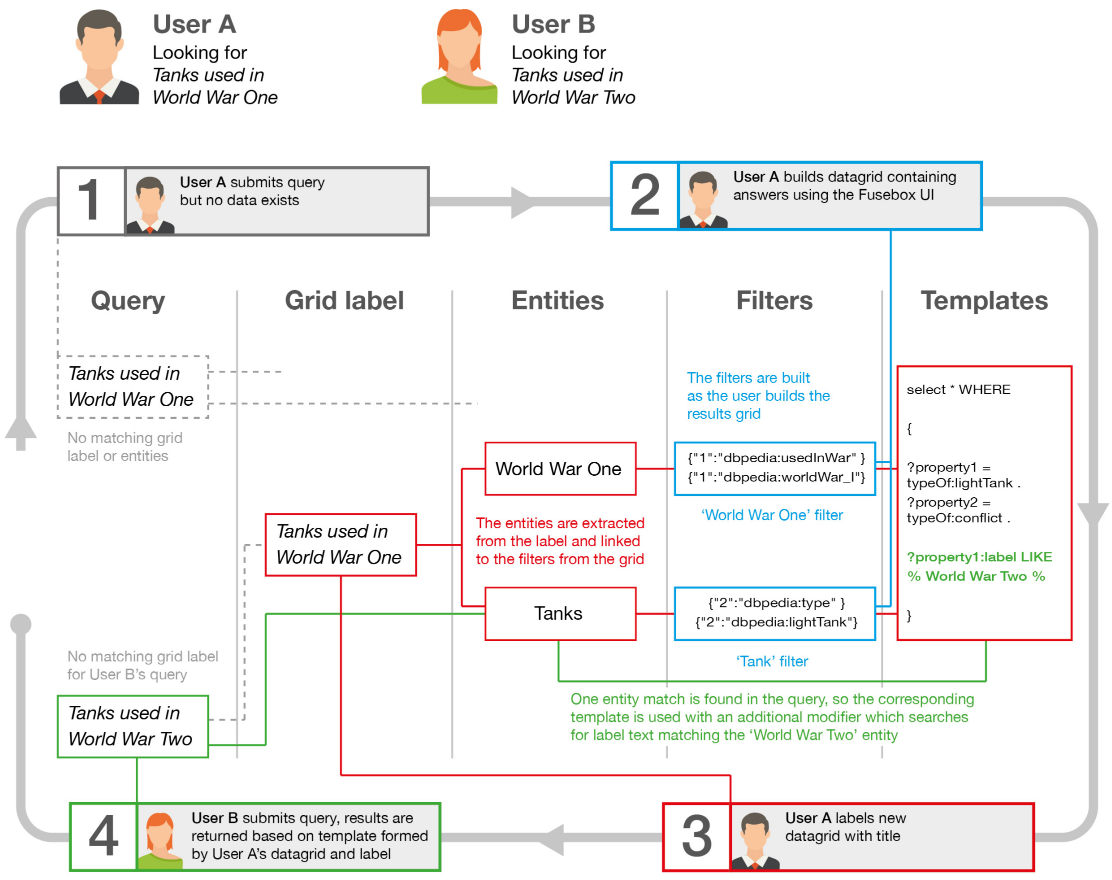

User A
Looking for Tanks used in World War One
User B
Looking for Tanks used in World War Two
1 User A submits query but no data exists
2 User A builds datagrid containing answers using the Fusebox UI
Query
Grid label
Entities
Filters
Templates
Tanks used in World War One
No matching grid label or entities
The filters are built as the user builds the results grid
select * WHERE
{
?property1 = typeOf:lightTank .
?property2 = typeOf:conflict .
?property1:label LIKE % World War Two %
}
World War One
{"1":"dbpedia:usedInWar" } {"1":"dbpedia:worldWar_I"}
'World War One' filter
Tanks used in World War One
The entities are extracted from the label and linked to the filters from the grid
Tanks
{"2":"dbpedia:type" } {"2":"dbpedia:lightTank"}
'Tank' filter
No matching grid label for User B's query
Tanks used in World War Two
One entity match is found in the query, so the corresponding template is used with an additional modifier which searches for label text matching the 'World War Two' entity
4 User B submits query, results are returned based on template formed by User A's datagrid and label
3 User A labels new datagrid with title

Figure 4: The AQUACOLD *Linked Data Feedback Loop*

4. Evaluation Methodology

AQUACOLD has been evaluated with a user evaluation and technical evaluation using 408 questions that comprise the 9th Question Answering Over Linked Data (henceforth QALD) challenge [39]. The challenge has run annually since its inception in 2011 and remains the most prominent evaluation series of its kind.

QALD-9 includes a wide range of questions ranging in complexity from simple (*When is Halloween?*) to advanced (*Give me the capitals of all countries that the Himalayas run through*). Each question is supplied with a gold standard set of answers and associated gold standard SPARQL query. Some questions are included that cannot be answered in the dataset, to show how the participating systems judge whether unanswerable questions represent a problem with the system itself, or lack of data.

DBpedia [40] is the knowledge base for these questions. Although DBpedia is large and wide ranging enough to cover a wide scope of questions, it is also incomplete with many ontology errors [39]. This is seen as a benefit, as it ensures competing LD-QA systems can handle incompleteness, modelling errors, missing property information and undefined entities.

Answers submitted by users using AQUACOLD in response to QALD-9 questions are evaluated using the standard measures of recall, precision and F1-Score against the gold standard provided in the QALD-9 dataset.

$$F_1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{1}$$

The **user evaluation** measured how successfully a random selection of AQUACOLD users with a range of backgrounds and proficiency could use the system to answer questions over the Linked Data web. On starting the experiment, each participant completed a short survey to capture their proficiency in: English; Web Search; Spreadsheets; SPARQL and overall IT skill. On submitting the survey a short video was presented instructing participants on how to use AQUACOLD. Each participant was then presented with a set of 5 random questions taken from the QALD-9 dataset and instructed to answer each using the AQUACOLD system.

The crowdsourced nature of the AQUACOLD system presents a challenge for the user evaluation as participants will be able to find answers faster and easier if the labels and filters that provide those answers have already been added to the system by other participants. To factor this into the evaluation, each of the 5 questions presented to participants were hosted on a different server (unbeknown to them) containing a different amount of preloaded data:

- **Server 1** has no **pre-existing data** available, participants can use the Query Builder (QB) only.
- **Server 2** has **all GS labels and filter sets** available through the Natural Language Search.
- **Server 3** has both correct and incorrect label and filters with **randomised vote scores**.
- **Server 4** has **labels and filter sets similar** to the GS which can be adapted with the QB
- **Server 5** has labels and filter sets **created by other users**, whether correct or incorrect.

By evaluating how participants answer questions when a variety of crowdsourced content is preloaded into the system, real world activity can be simulated and a reliable evaluation obtained.

The **technical evaluation** measured the maximal performance of AQUACOLD in answering all questions from the QALD-9 dataset and compares the system's results against competing systems in the QALD-9 challenge.

This evaluation was carried out by an AQUACOLD system expert (the system developer) attempting to answer each of the 408 questions, negating the chance that lack of familiarity with the system would impact the precision or recall of each answer. For the comparative analysis between AQUACOLD and other QALD-9 systems, precision, recall and f-score are recorded for AQUACOLD and compared against those reported for competing systems.

Finally, an evaluation of the **template coverage** available to AQUACOLD was included as a measurement of the total number of queries the system could answer over DBpedia when seeded with templates from the QALD-9 dataset. This is used to assess how expeditiously the system expands its query coverage over time.

5. Results

5.1. User Evaluation Results

30 participants took part in the user evaluation after signing up online and completing the entry questionnaire. This exceeds the minimum sample of size of 16 put forward by TREC [41] but was lower than the number anticipated.

Each participant was assigned 5 random questions from the QALD-9 dataset to answer sequentially using AQUACOLD (see section 4), resulting in 150 questions answered in total. Questions that were identified as unanswerable or only partially answerable in the technical evaluation (see 5.3.1) were removed, leaving only questions that were fully answerable according to the supplied gold standard answers.

77 questions assigned to participants were answered correctly, retrieving all correct answers (52% of the total), 46 answered incorrectly with no correct answers (31% of total), 11 answered partially correctly retrieving some but not all of the correct answers (7% of total) and 15 questions were abandoned when the participant moved on to the next question without answering (10% of total). Precision, recall and f-measure were calculated for all attempts, resulting in a mean average score of **0.52 precision**, **0.56 recall** and **0.52 F1 score**.

Analysis of average F1 score per question in the order the presented to participants (table 1) shows a slight increase (0.08 in F1 Score) in average participant performance after Q1, which may indicate participants are getting used to the system as they progress through the experiment.

5.2. User Results Per Server

Figure 5 highlights that participants answered question on Servers 1, 3 and 4 with a similar rate of success, which is surprising given that Servers 3 and 4 contain some natural language labels that can help participants answer some questions. The similarity in results between these servers indicates that participants found the extra work required to identify which of the 3 labels is correct (for Server 3) or to adapt the partially correct labels (for Server 4) were no more useful for answering the question than using the Query Builder exclusively.

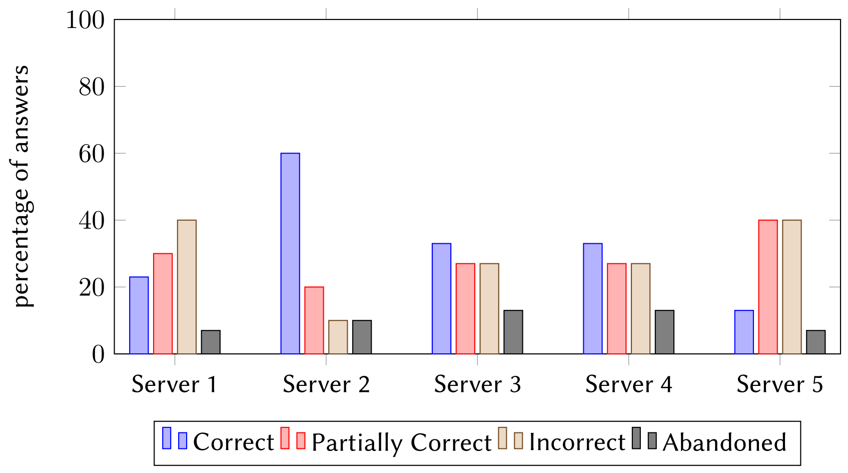


Figure 5: Percentage of correct / partially correct / incorrect / abandoned questions per server

Server 1 contained no label or filter set data, requiring the participant to use the Query Builder to answer questions. The majority of participants on this server (75%) attempted to find the question answer using the natural language search tool initially and when no answer was found, attempted to find the answer using the Query Builder. The next action most common at this stage was to search for a relevant entity in the subject filter position (48% of users), regardless of whether the entity should have been a subject or propertyValue. This pattern is consistent for all servers, indicating that participants may not have grasped the differences between filter types in the tutorial.

Server 2 contained gold standard labels and corresponding filter sets available via the natural language search tool. This server shows a significant increase in the percentage of correct participant answers (60%), which is to be expected as all participants have to do is type the question into the search tool and choose the suggested label (of which there is only one) in the search results. This demonstrates the utility of AQUACOLD in an idealised scenario, where the system has been seeded with only correct answers by the crowd.

Server 3 contained one correct label and filter set and two incorrect label and filter sets for each question, with each answer assigned a random visible confidence score from -5 to +5. The majority of participants (83%) initially chose the matching label with the highest vote score

Question Order	Avg. % of correct answers	Avg. Precision	Avg. Recall	Avg. F1 Score
1	45%	0.42	0.45	0.43
2	51%	0.50	0.51	0.51
3	66%	0.66	0.66	0.64
4	59%	0.51	0.59	0.52
5	56%	0.53	0.56	0.53

Table 1
Average results for all questions in the order presented to participants

when searching for an answer via natural language search. Of the participants who selected a label which resulted in incorrect answers, (26%) used the query builder instead to try and find an answer to the question, with 74% returning to the list and selecting a different option. This indicates that participants were willing to explore the results returned by selecting a lower voted label and could be further ameliorated by encouraging users to exercise their own votes to redress incorrectly assigned labels.

Server 4 contained labels and associated filter sets related, but not identical to, the gold standard answer and required editing with the removal of one filter and the addition of another. 54% of participants on Server 4 searched for then selected a related label when searching for the answer to their assigned question rather than using the query builder interface. Of these, 100% edited the filter set returned, of which 71% successfully reformulated the filter set to find the correct answer. This indicates that participants were comfortable with tweaking existing filter sets by editing them to find the correct answer and suggests that system utility extends beyond scenarios where only the completed answer is available.

Server 5 contained labels and filter sets populated by previous participants when using this server. Participants on this server recorded the lowest number of correct answers (13%) and the joint highest number of incorrect answers (40%) which is surprising, given that the natural language labels and associated filter sets created by previous participants should have helped users answer the question. One explanation may be that the low number of overall participants (30) did not allow enough opportunity for sufficient, quality, crowdsourced results to influence the overall score for this server.

The use of multiple preloaded servers to mimic the effects of distinct crowdsourced conditions on AQUACOLD user interactions has shown that the system performs well when one correct answer has been entered by the crowd or a similar answer is available that can be refined using the grid controls. In situations where no answer is available, or multiple answers have been added by the crowd with misleading vote scores, the system performs less well. More testing is needed with a larger sample size to get an accurate measure of how the crowdsourced component performs at scale.

5.3. Technical Evaluation

5.3.1. Query Coverage

To identify the maximum query coverage available to AQUACOLD, an expert user (the system developer) attempted to answer all 408 QALD-9 questions. Of these, 342 were fully answerable, 54 were unanswerable and 12 were partially answerable.

To be classified fully answerable, the answers returned by AQUACOLD must match the gold standard answers supplied for the associated question by QALD-9. To be classified partially answerable, at least one returned answer must match at least one answer for the associated question from QALD-9. To be classified unanswerable, no returned answers match any supplied by QALD-9 for that question.

The 66 unanswerable or partially answerable questions could become answerable if the AQUACOLD UI was developed further to incorporate controls for the required SPARQL elements (see table 2). Details of potential future developments can be found in section 6.

Reason	**# questions affected**
Requires Union	23
Requires additional ontology	9
Requires aggregation with GT / LT	9
Requires relative reference	8
Requires aggregation with LIMIT	6
Requires cell ungrouping	5
Result set too large	5
Requires text substring search	1
Requires non English translation	1

Table 2
Reasons for unanswerable questions in AQUACOLD (some included multiple unanswerable elements)

5.3.2. Template Coverage

AQUACOLD's query templating feature provides answers to multiple questions from a single template. We evaluate the size of templates produced by each QALD-9 question to measure the template coverage offered by AQUACOLD. Our sample size is limited to 342 questions - the number of QALD-9 questions answerable by the system (see 5.3.1).

Templates could be produced from 227 of the 342 answerable questions. A total of 839,938 queries were produced from the 227 templates, giving an average of 3,700 queries produced for each templateable question.

Where templates were unable to be produced from a query, this was caused by the system being unable to calculate a match between words used in the query label and the labels used in the filter sets. For example, the query *"Give me all Taikonauts"* requires the filter sets **occupation**:*Astronaut* and **nationality**:*Russian*. There are no shared words between the query *"Give me all Taikonauts"* and these labels, so a template cannot be produced. If the query label was instead *"Give me all Russian Astronauts"*, templates could be produced that answer queries such as *"Give me all Chinese Astronauts"* and *"Give me all Russian Singers"*.

This evaluation demonstrates that AQUACOLD templating system can vastly increase the system's available query coverage, rapidly increasing its utility as a Linked Data Question Answering tool.

5.3.3. AQUACOLD performance in QALD-9 compared to competing systems

To evaluate AQUACOLD using consistent metrics to similar systems that took part in the QALD-9 challenge, all 408 questions were attempted in AQUACOLD by an expert user (the system developer) with the resulting answers converted into the established IR metrics of precision, recall and f-measure using the formula detailed in section 4.

Mean average system recall for all 408 QALD-9 training questions was recorded as **0.87**, with a mean average system precision of **0.9**, giving an overall system f-measure score for AQUACOLD of **0.88**.

	WDAqua	gAnswer 2	TeBaQA	Elon	QASystem	AQUACOLD
System f-score	0.29	0.43	0.22	0.10	0.20	**0.88**

Table 3
Comparison of system f-scores from QALD-9 participating systems [42] against AQUACOLD

AQUACOLD has a higher f-score than competing systems (see table 3[1]), demonstrating that more questions can be successfully answered from QALD-9 dataset than its competitors.

It could be argued that this higher score is due to the use of the Query Builder component which allows precise selection of the URIs, filter sets and modifiers and is incomparable with competing systems that use a 'pure natural language' approach. However, as subsequent users can retrieve answers to these questions (and any similar ones created by the templating system) using natural language, with fuzzy matching for both syntax and entities, we argue that the comparison against other pure natural language approaches is applicable, once enough data has been seeded, as the result for the end user is the same - they are provided with answers from the linked data web in response to a natural language query.

6. Summary and Future Work

AQUACOLD is a novel question answering tool that combines natural language search, a faceted browsing interface and a crowdsourced templating system to answer complex questions from the Linked Data web. Our evaluation has demonstrated that the system can answer a wide variety of question types accurately and that the templating system can vastly expand the answerable question space based on a small number of templates seeded by users.

The evaluation has demonstrated that most participants can find the correct answer using the system's natural language search system if it has been seeded by the crowd previously and can use the grid controls to refine partially correct filter sets to produce the correct answer. Participants are less successful when filtering answers incorrectly labelled by other users or finding answers using the query builder component, but users with greater experience of spreadsheets or structured data tools perform better at this task. This suggests a potential implementation of the system where technically skilled users create filter sets using the query builder for less technically able users to find using the natural language search tool.

Future work will include rerunning the evaluation with a larger cohort size to return more robust results that should highlight how larger volumes of organically seeded data affect the user experience. We plan to expand the number of available SPARQL operators supported by the system to enable more complex queries involving unions, multi-language support, relative references and additional ontologies other than DBPedia to be answered. Further work will also include investigating whether the faceted query builder component could be replaced with a natural language question-and-answer based system, negating the need for the user to differentiate between subject, property and property value filters which less technically able participants reported confusion over.

[1]Elon and QASystem not yet published

References

[1] H.-J. Oh, K.-Y. Sung, M.-G. Jang, S.-H. Myaeng, Compositional question answering: A divide and conquer approach, Information Processing & Management 47 (2011) 808–824. doi:10.1016/j.ipm.2010.03.011.

[2] M. S. Bernstein, J. Teevan, S. Dumais, D. Liebling, E. Horvitz, Direct answers for search queries in the long tail, Proceedings of the 2012 ACM annual conference on Human Factors in Computing Systems (2012) 237–246. URL: http://dl.acm.org/citation.cfm?doid=2207676.2207710. doi:10.1145/2207676.2207710.

[3] R. W. White, M. Bilenko, S. Cucerzan, Studying the Use of Popular Destinations to Enhance Web Search Interaction, Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '07 (2007) 159–166. URL: http://dl.acm.org/ft_gateway.cfm?id=1277771&ftid=437536&dwn=1&CFID=739208913&CFTOKEN=18406667. doi:10.1145/1377488.1377490.

[4] A. Singhal, Introducing the knowledge graph: things, not strings, Official google blog 5 (2012).

[5] T. Berners-Lee, The Great Unveiling, 2009. URL: http://www.ted.com/index.php/talks/tim_berners_lee_on_the_next_web.html>.

[6] W3c, Linked Data definition, 2015. URL: http://www.w3.org/standards/semanticweb/data.

[7] A. Bozzon, M. Brambilla, S. Ceri, P. Fraternali, M. Dipartimento, V. Ponzio, Liquid Query : Multi-Domain Exploratory Search on the Web (2010) 161–170.

[8] S. Shekarpour, A. C. N. Ngomo, S. Auer, Query segmentation and resource disambiguation leveraging background knowledge, CEUR Workshop Proceedings 906 (2012) 82–93.

[9] K. Höffner, J. Lehmann, Survey on Challenges of Question Answering in the Semantic Web Survey on Challenges of Question Answering in the Semantic Web 0 (2016).

[10] H.-j. D. C.-y. Wu, R. T.-h. Tsai, From Entity Recognition to Entity Linking : A Survey of Advanced Entity Linking Techniques, The 26th Annual Conference of the Japanese Society for Artitifical Intelligence (2012) 1–10.

[11] D. Damljanovic, J. Petrak, M. Lupu, H. Cunningham, M. Carlsson, G. Engstrom, B. Andersson, Random Indexing for Finding Similar Nodes within Large RDF graphs (2012) 1–15.

[12] A. Kittur, J. V. Nickerson, M. Bernstein, E. Gerber, A. Shaw, J. Zimmerman, M. Lease, J. Horton, The future of crowd work, in: Proceedings of the 2013 conference on Computer supported cooperative work, ACM, 2013, pp. 1301–1318.

[13] B. F. Green, A. K. Wolf, C. Chomsky, K. Laughery, Baseball - an aautomatic question answerer (1961).

[14] W. A. Woods, R. Kaplan, Lunar rocks in natural English: Explorations in natural language question answering, Linguistic structures processing 5 (1977) 521–569.

[15] I. Androutsopoulos, G. Ritchie, P. Thanisch, Masque / sql . An Efficient and Portable Natural Language Query Interface for Relational Databases (1995) 1–7.

[16] A. M. Gliozzo, O. Biran, S. Patwardhan, K. McKeown, Semantic Technologies in IBM Watson TM, Proceedings of the Fourth Workshop on Teaching NLP (2013) 85–92.

[17] M. Damova, D. Dannells, Natural language interaction with semantic web knowledge bases and lod, … Semantic Web. … (2013) 1–15. URL: http://www.molto-project.eu/sites/

default/files/bookchap_0.pdf.
[18] J.-D. Kim, K. B. Cohen, Natural language query processing for SPARQL generation: A prototype system for SNOMED-CT, Proceedings of {BioLINK SIG} 2013 (2013) 32–38. URL: https://sites.google.com/site/biolinksig2013/program/biolinksig2013_Kim_Cohen.pdf.
[19] A. Russell, P. R. Smart, D. Braines, N. R. Shadbolt, NITELIGHT: A graphical tool for semantic query construction, CEUR Workshop Proceedings 543 (2009) 1–10.
[20] E. Kaufmann, A. Bernstein, Evaluating the usability of natural language query languages and interfaces to Semantic Web knowledge bases, J. Web Sem. 8 (2010) 377–393.
[21] H. I. Storage, R. Content, Falcons : Searching and Browsing Entities (2008) 1101–1102.
[22] D. Huynh, D. Karger, Parallax and companion: Set-based browsing for the data web, WWW Conference (2009) 2005–2008. URL: http://davidhuynh.net/media/papers/2009/www2009-parallax.pdf.
[23] T. Berners-lee, Y. Chen, L. Chilton, D. Connolly, R. Dhanaraj, J. Hollenbach, A. Lerer, D. Sheets, Tabulator : Exploring and Analyzing linked data on the Semantic Web, Proceedings of the 3rd International Semantic Web User Interaction Workshop (2006).
[24] P. Fafalios, Y. Tzitzikas, X-ENS: Semantic Enrichment of Web Search Results at Real-time, Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval (2013) 1089–1090. doi:10.1145/2484028.2484200.
[25] S. Lohmann, S. Dietzold, Interacting with Multimedia Content in the Social Semantic Web, Design (2008).
[26] D. Tunkelang, Faceted search, Synthesis lectures on information concepts, retrieval, and services 1 (2009) 1–80.
[27] M. Arenas, B. Cuenca Grau, E. Kharlamov, Š. Marciuška, D. Zheleznyakov, Faceted search over RDF-based knowledge graphs, Journal of Web Semantics 37-38 (2016) 55–74. doi:10.1016/j.websem.2015.12.002.
[28] S. Ferré, Expressive and Scalable Query-Based Faceted Search over SPARQL Endpoints Sébastien Ferré To cite this version : HAL Id : hal-01100313 Expressive and Scalable Query-based Faceted (2015).
[29] P. Hoefler, M. Granitzer, E. Veas, C. Seifert, Linked data query wizard: A novel interface for accessing sparql endpoints, CEUR Workshop Proceedings 1184 (2014).
[30] M. Van, M. V. Kleek, B. Moore, D. Karger, heterogeneous information sources on the web Citation Accessed Citable Link Detailed Terms Atomate It ! End-user Context-Sensitive Automation using Heterogeneous Information Sources on the Web (2017).
[31] E. Kaufmann, A. Bernstein, Evaluating the Usability of Natural Language Query Languages and Interfaces to Semantic Web Knowledge Bases, SSRN Electronic Journal (2010). URL: https://www.ssrn.com/abstract=3199491. doi:10.2139/ssrn.3199491.
[32] D. Ferrucci, E. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. Prager, others, Building Watson: An overview of the DeepQA project, AI magazine 31 (2010) 59–79.
[33] C. Sarasua, E. Simperl, N. F. Noy, A. Bernstein, J. M. Leimeister, Crowdsourcing and the Semantic Web: A Research Manifesto, Human Computation 2 (2015) 3–17. URL: http://hcjournal.org/ojs/index.php?journal=jhc&page=article&op=view&path[]=45. doi:10.15346/hc.v2i1.2.
[34] M. Acosta, E. Simperl, F. Flöck, M.-E. Vidal, HARE: A hybrid SPARQL engine to enhance

query answers via crowdsourcing, in: Proceedings of the 8th International Conference on Knowledge Capture, ACM, 2015, p. 11.
[35] G. Demartini, D. E. Difallah, P. Cudré-Mauroux, ZenCrowd: Leveraging Probabilistic Reasoning and Crowdsourcing Techniques for Large-Scale Entity Linking, Proceedings of the 21st international conference on World Wide Web - WWW '12 (2012) 469–478. URL: http://dl.acm.org/citation.cfm?id=2187836.2187900%5Cnhttp://www2012.wwwconference.org/proceedings/proceedings/p469.pdf. doi:10.1145/2187836.2187900.
[36] J. Fan, M. Zhang, S. Kok, M. Lu, B. C. Ooi, CrowdOp: Query optimization for declarative crowdsourcing systems, 2016 IEEE 32nd International Conference on Data Engineering, ICDE 2016 (2016) 1546–1547. doi:10.1109/ICDE.2016.7498417.
[37] G. Demartini, B. Kraska, M. Franklin, CrowdQ: Crowdsourced Query Understanding, Conference on Innovative Data Systems Research (CIDR) (2013) 4. URL: http://www.cidrdb.org/cidr2013/Papers/CIDR13_Paper137.pdf.
[38] D. Crow, Google Squared: web scale, open domain information extraction and presentation, in: European Conference on Information Retrieval, Industry Day, 2010.
[39] P. Cimiano, C. Unger, A. Freitas, Question Answering over Linked Data, 10th Reasoning Web Summer School 0 (2011) 3–13. URL: http://www.websemanticsjournal.org/index.php/ps/article/view/339.
[40] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, S. Hellmann, DBpedia - A crystallization point for the Web of Data, Web Semantics: Science, Services and Agents on the World Wide Web 7 (2009) 154–165. URL: http://linkinghub.elsevier.com/retrieve/pii/S1570826809000225. doi:10.1016/j.websem.2009.07.002.
[41] S. T. Dumais, N. J. Belkin, The TREC interactive tracks: Putting the user into search, TREC: Experiment and evaluation in information retrieval (2005) 123–152.
[42] R. Usbeck, R. H. Gusmita, M. Saleem, 9th Challenge on Question Answering over Linked Data (QALD-9) (2018).